(Check) 26 - Modeling Software Effort Estimation Using Hybrid PSO-ANFIS

by Arta Moro Sundjaja

FILE	-_MODELING_SOFTWARE_EFFORT_ESTIMATION_USING_HYBRID_PSO-ANFIS.PDF (1.06M)		
TIME SUBMITTED	06-APR-2017 04:14PM	WORD COUNT	3176
SUBMISSION ID	649646071	CHARACTER COUNT	17245

2016 International Seminar on Intelligent Technology and Its Application

Modeling Software Effort Estimation Using Hybrid PSO-ANFIS

Suharjito
Magister of Information Technology
Bina Nusantara University
Jakarta, Indonesia
suharjito@binus.edu

Saka Nanda
Magister of Information Technology
Bina Nusantara University
Jakarta, Indonesia
sakatecstar@yahoo.co.id

Benfano Soewito
Magister of Information Technology
Bina Nusantara University
Jakarta, Indonesia
bsoewito@binus.edu

***Abstract*— Accurate estimating software development effort is essential in effective project management processes such as budgeting, project planning and control. To achieve an accurate estimate some algorithmic estimation techniques proposed to eliminate or reduce inaccuracies estimation. COCOMO is a parametric model used to estimate software effort. However, so far no model has proven successful to effectively and consistently predict software effort. Parametric models are considered vulnerable when faced with the problem of non-linearity of the complex in the parameters. In recent years, some estimation technique appears using intelligent systems to predict software effort. This study uses a model Neuro-fuzzy optimized with PSO to get the right model to improve the estimation effort at NASA dataset software project. Parameter cost driver, consisting of 17 feature COCOMO will then be optimized using PSO techniques to get a better prediction accuracy. Furthermore, the results of the optimization will be trained in using the algorithm to get a prediction Neuro-fuzzy effort. The performance of the proposed estimation model will be evaluated with some other intelligent system model parameters to evaluate several criteria such as Mean Standard Error (MSE), Mean Magnitude of Relative Error (MMER), and Level Prediction (Pred). The model that best shows the error rate MSE and MMER lowest to highest Pred.**

Keywords—Effort Estimation; ANFIS; COCOMO; Particle Swarm Optimization

I. INTRODUCTION

The development of the software industry is very rapid in recent causes the cost of the software to be one of the topics that interest [1]. Estimates of the cost to be one measure of success in a software project. An accurate estimate for the software effort, cost and scheduling is very important to manage financial issues and monitor the activities of development and on-time delivery. Based on data from the Standish Group's CHAOS Report 2012 [2] EXTREME, 30,000 software projects, 39% of project failures, 43% experienced problems, only18% of successful projects. In addition to financial losses, the company's IT project failure caused a decline in the company's reputation. To achieve an accurate estimate, many contributions proposed and validated estimation techniques to reduce and eliminate inaccuracies estimation. Highlights of this research is to design software evaluation effort using Adaptive Neuro-Fuzzy Inference System (ANFIS) the historical dataset COCOMO 81 and shows the significance of this technique compared to other machine learning techniques. This study will use a blend of techniques PSO with ANFIS tested for applicability to predict COCOMO feature. Two methods will be compared in terms of the accuracy of their predictions. The remainder of the paper is divided in different sections as follows: Section II includes a brief literature review about the concepts and techniques used in current model. Section III presents the proposed model based on ANFIS Optimization with PSO for software cost estimation. Experiments and results are described in section IV and conclusion of the paper is described in section V and in the last; Appendix shows the comparison between those models.

II. LITERATURE REVIEW

In four or five decades, there are many proposed estimation technique, referred to as the estimation model. To solve this problem, there are various methods of machine learning [3, 4] [5]. One of the methodological variations among them, Artificial Neural Networks [6, 7, 8], Fuzzy Logic [9, 10], Evolutionary Computing such as Genetic Algorithm [11] and Particle Swarm Optimization [12, 13, 14]. This method is suitable to solve real-world ambiguity. Artificial neuro-fuzzy inference systems have been applied in software effort estimation fields. Many studies on software effort prediction using artificial neuro-fuzzy inference system [15, 13, 16, 8] have been realized recently. Most of them [6] use a gradient descendent technique for optimizing the antecedent parameters and a least means square method for the consequent parameters of the ANFIS. More recently [17, 11] an optimization technique based on genetic algorithm was proposed for training the parameters in the antecedent part of a fuzzy system.

A. COCOMO

Various parametric models have sprung up, such as COCOMO [18] COCOMO II [19], SLIM [20], ESTIMACS [21], all based Functional size Measurement (FSM) [22] in the estimation of effort. The quality of the data for this model bias depend son subjective assessments for each of the functional size. Model on COCOMO and SLIM depends on the number of SLOC (Source Lines of Code) to be estimated before starting the effort estimation process. COCOMO (Constructive Cost Model) is a regression model developed by Prof. Barry W. Boehm [19]. This method introduces a non-linear approach in 1981. COCOMO categorizes projects into three levels, namely Basic, Intermediate, and Detail. When the data set is still widely used in the prediction of effort and budget planning software [15]. COCOMO II model can be described by the following equation:

978-1-5090-1709-6/16/$31.00 ©2016 IEEE

$$Effort = A \times (Size)^B \times \prod_{i=1}^{17} EM_i \quad (1)$$

Where A and B are constants initial calibration, Size refers to the size of software project measured Source Lines of Code (KSLOC), SF is a scale factor, and EM is Effort multiplier. COCOMO II has 17 Effort Multiplier (EMS) used in the model Post-Architecture.

B. Fuzzy Membership Function

Several types of membership functions including: Triangular, Gaussian bell, trapezoidal, sigma, S function and the function of Z The three of them namely Triangular, Gaussian and trapezoidal very commonly used in software estimation model as more appropriate to describe the linguistic term.

$$\mu_T(x) = \begin{cases} \frac{x-l1}{m-l1}, & x \in [l1, m] \\ \frac{l2-x}{l2-m}, & x \in [m, l2] \\ 0, & x \notin [l1, l2] \end{cases} \quad (2)$$

Where l1 is the left boundary value of membership, mk is the value of capital and l2 is the right limit membership value (l1 <m <l2).

$$\mu_T(x) = \begin{cases} \frac{x-l1}{l2-l1}, & x \in [l1, l2] \\ 1, & x \in [l2, l3] \\ \frac{l2-x}{l2-m}, & x \in [l3, l4] \\ 0, & x \notin [l1, l4] \end{cases} \quad (3)$$

Wherel1is the left boundary value of membership, l2 and l3 and l4 is a middle limit is the right limit membership value (l1 <l2 <l3 <l4).

C. ANFIS

ANFIS is an integrated model between Fuzzy Inference System (FIS) with Neural Network. Keys to success is finding the FIS rule base. FIS convert human knowledge into a rule base in order to maximize performance and minimize the model error output.

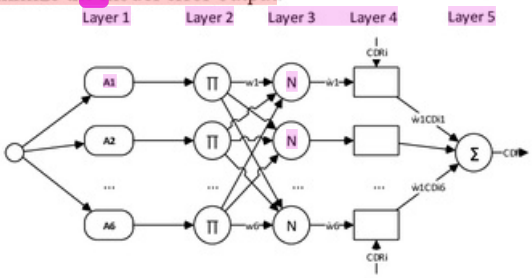

Fig. 1 ANFIS Structure

The function of each node is described as follows: 1. Layer1: each node k in a square layer node with the function:

$$O_k^1 = \mu_{ik}(x) \quad (4)$$

Where x is the input of node k, and A_{ik} is a fuzzy set of rating the node function. O_k^1 is a membership function hat determines the degree of membership A_{ik}. 2. Each node labeled Π is a multiplier value. 3. Layer 3: Normalized firing strength. Each node labeled N which is calculated by taking the ratio of the k-th rule firing strength of the overall total of all rules firing strength. 4. Layer 4: Setting the consequent parameters. Each node k in a square layer node with function:

$$O_k^4 = \bar{w}_k f_k = \bar{w}_k CD_{ik} \quad (5)$$

Layer 5: There is a single node with symbol Σ which calculates the overall output of the incoming signal. This stage is also called defuzzification.

D. PSO

PSO is a heuristic method aims to optimize iterative problem with trying to improve the candidate solutions linked to certain measures of quality. The method is known as Meta Heuristic [23]. PSO is a population-based search where each solution is represented as a particle of the population (called swarm). Each particle should be noted that the optimal solution path through them, the solution is called locally optimal solution (Pbest). Each particle should also have social behavior of each particle to find the optimal solution in the current search, the so-called global optimal solution (Gbest).

$$v_{id} = w \times v_{id} + c_1 \times Rand() \times (p_{id} - x_{id}) + c_2 \times Rand() \times (p_{gd} - x_{id}) \quad (6)$$

When the particles get Pbest and Gbest, using the above equation updating velocity of the displacement of each particle. Particle next coordinates for the current location coordinates are updated as low speed add the new coordinates.

III. Methodology

In this study, Pearson Correlation was used to analyze linear relationship between the 15th cost drivers with the actual effort. The coefficient 'r' is calculated by the following equation:

$$r = \frac{\Sigma(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\Sigma(x_i - \bar{x})^2 (y_i - \bar{y})^2}} \quad (7)$$

Where X is the cost factor and Y is the actual effort. $\bar{x}$ is the average value of the cost factor of 63 data. Similarly $\bar{y}$ is the average actual effort of 63 Data. Figure 2 shows the correlation between the cost factors with the actual effort by COCOMO 81 dataset.

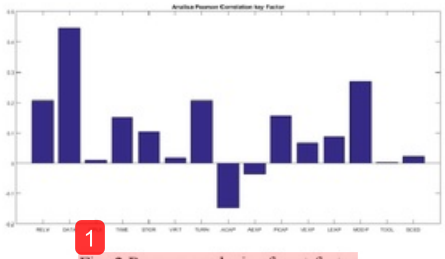

Fig. 2 Pearson analysis of cost factor

Through the Pearson Correlation analysis, a relatively strong positive correlation (0.449) was found in factor Database Size (DATA) and the actual effort. This would suggest if the size of the database increases, the value of effort also increases. Likewise, linear associations were also found at factor cost MODP, RELY, and TURN to the actual effort. Furthermore, one-way ANOVA was used to test whether it

is true for variables are the same means. This is done to determine whether there are important differences between the means of two unrelated groups, namely the cost factor and the actual group effort. Null hypothesis is made means of two variables are the same.

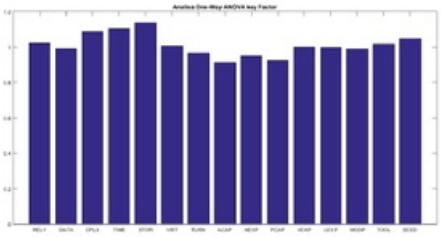

Fig. 3 one way Anova analysis of Cost Factor

Factors to be selected from this process is ACAP, AEXP, PCAP and LEXP. Thus eight dominant features are selected will be processed in training.

In this study, ANFIS uses PSO to adjust the parameters of the membership function. Membership function tested in this study are triangular and Gaussian The algorithm used in the approach described in Figure5.Featuresof data that all seven dominant cost factor and one LOC preprocessing results incorporated into the ANFIS (see Fig. 2). in first stage, we conducting training ANFIS with data from the previous stage. The training process allows the system to adjust the parameters as input/output are included. The training process stops when the number of epoch is reached or the number of error-rates achieved. The second stage: create a vector with the number of dimensions N where N is the number of membership functions. Vector contains the parameters of membership functions and will be optimized by PSO algorithm. Mean-Squared Error is defined as fitness function. The third stage: Determine the initiation parameter PSO shown in Table I. Parameters randomly initiated in the first phase and then updated by PSO algorithm. The fourth stage: Parameter produced PSO then extracted to output ANFIS. The output is an effort predictions of PSO-ANFIS approach.

TABLE I INITIAL PARAMETER PSO

Parameter	Value
Particle	25
Iteration	2000
Cognitive Acceleration	2.0
Social Acceleration	2.0
Inertia weight ω_{min}	0.9
Final Inertia weight ω_{max}	0.4

Parameters are initialized at random in the first stage and then updated using the PSO algorithm. In each iteration, one of the parameters of the membership function will be updated.

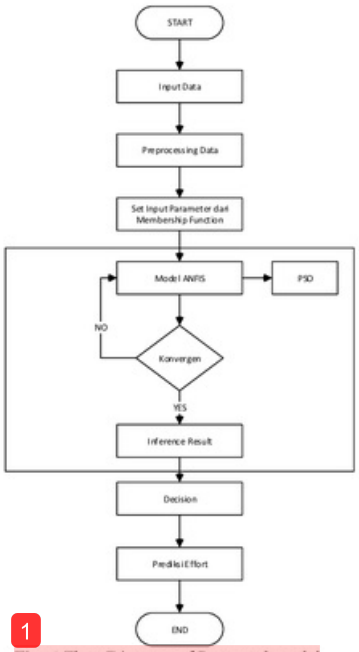

Fig. 4 Flow Diagram of Proposed model

IV. RESULTS AND DISCUSSION

Evaluate the accuracy of estimation can be done by comparing the results of the production effort and the actual effort to calculate the MSE (Mean Squared Error) and MRE (Magnitude of Relative Error) [24]. The average MER has better results than the average MRE (MMRE). MSE can be described by the following equation where i is the number of observations:

$$MSE = \left(\frac{1}{n}\sum_{i=1}^{n}(P_i - A_i)^2\right) \quad (8)$$

MER can be described by the following equation where i is the number of observations:

$$MER_i = \frac{|Actual\ Effort_i - Estimated\ Effort_i|}{Estimated\ Effort_i} \quad (9)$$

Pred(l) is used as a complement to the criteria for calculating the percentage estimates are under l to actual effort. Generally, the valueusedforlwas25%.

$$Pred(x) = (1/n)\sum_{i=1}^{n}\begin{cases}1, if\ MER \le x\\ 0, if\ MER \ge x\end{cases} \quad (10)$$

A. Triangular Membership Result

For standard ANFIS models, training results indicate over fitting if using110 epoch. Best fitting was found in the number of training10 epoch. Error training is lowest at 4.94. While ANFIS-PSO training results showed a decrease in error until 2.22 in the epoch to 107. Since the model used is Partition Grid, then the number of rules generated is 256 rules.

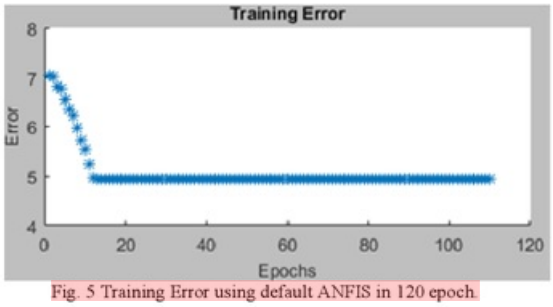

Fig. 5 Training Error using default ANFIS in 120 epoch.

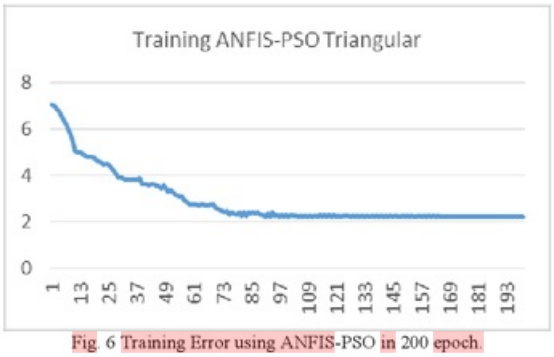

Fig. 6 Training Error using ANFIS-PSO in 200 epoch.

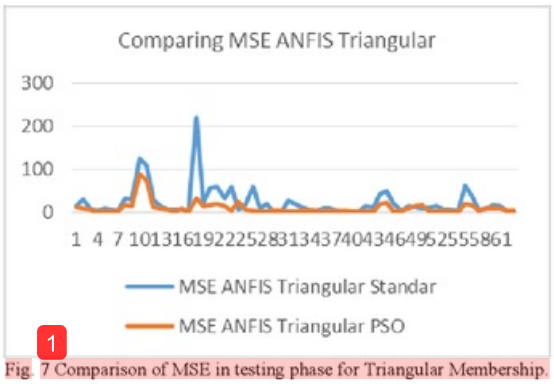

Fig. 7 Comparison of MSE in testing phase for Triangular Membership.

In the model of ANFIS Triangular, the average MSE ANFIS standards using back propagation optimization is 22.61 while the average MSE with ANFIS modification with optimization PSO is 9.30.

B. Gaussian Membership Result

Training results indicate over fitting if using 500 epoch. Best fitting was found in the number of training 20 epoch. Error lowest at training is 8.15. While on ANFIS models PSO error rate can still be decreased after the epoch to 500. The lowest error when training is 7.04. Because the model used is Partition Grid, then the number of rules generated is 256 rules.

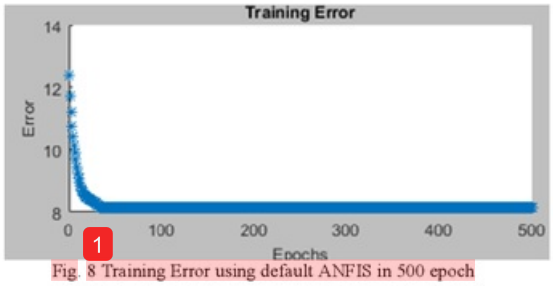

Fig. 8 Training Error using default ANFIS in 500 epoch

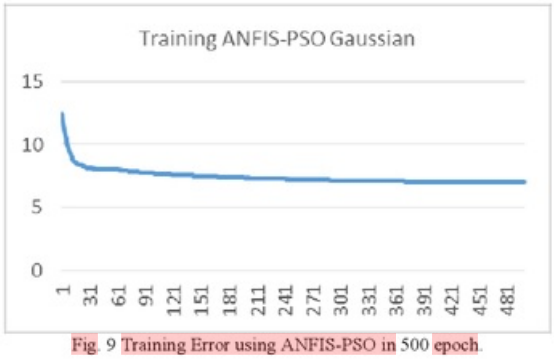

Fig. 9 Training Error using ANFIS-PSO in 500 epoch.

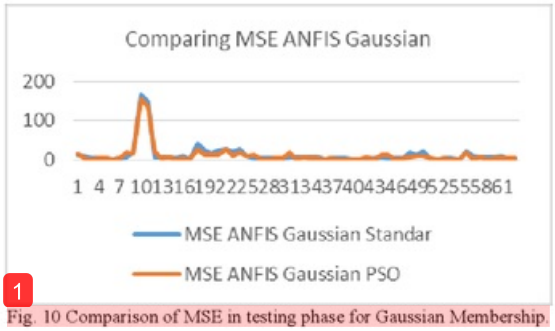

Fig. 10 Comparison of MSE in testing phase for Gaussian Membership.

In ANFIS Gaussian models, the average MSEANFIS standards using back propagation optimization is 12:48 while the average MSE with ANFIS modification with optimization PSO is 10.63.

C. Subtractive Clustering Result

To ANFIS standards, training results indicate over fitting if using 150 epoch. Best fitting standard ANFIS was found in the number of training 2 epoch. Error training is lowest at 0:52. As for the best-fitting ANFIS PSO was found in the number of training 150 epoch. Lowest error when training is 0:07. Rule number generated by the model are the Subtractive Clustering are 40 rules.

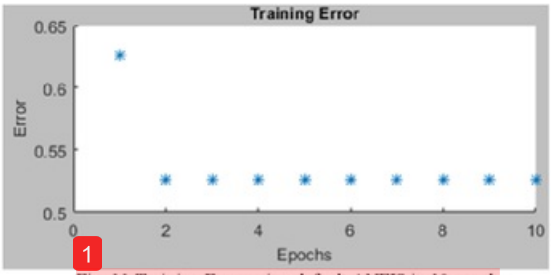

Fig. 11 Training Error using default ANFIS in 10 epoch.

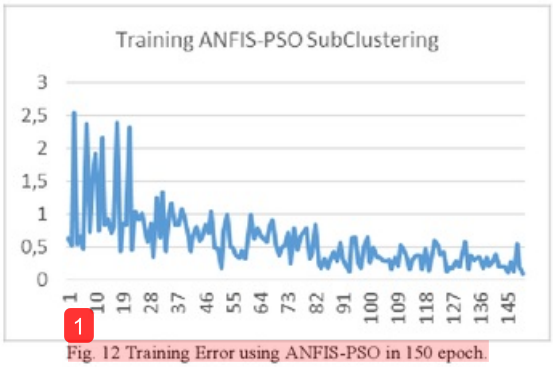

Fig. 12 Training Error using ANFIS-PSO in 150 epoch.

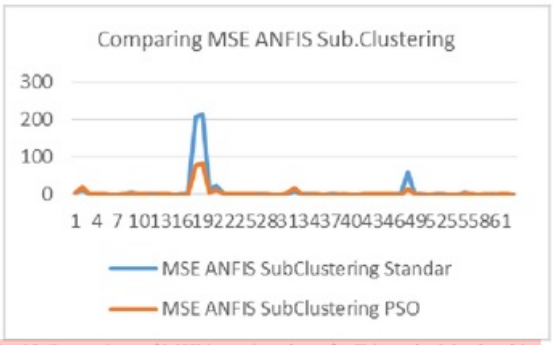

Fig. 13 Comparison of MSE in testing phase for Triangular Membership.

In Sub .Clustering ANFIS models, the average MSE ANFIS standards using back propagation optimization is 9.31 while the average MSE with ANFIS modification with optimization PSO is 4.09.

V. Conclusion and Future Works

Based on experiment conducted in Table II, it can be concluded that ANFIS-PSO model evaluation shows that the PSO implemented optimization results better estimate the level MMRE, MSE lower and higher Pred than the previous ANFIS approach. In addition, the number of epoch used during training are also shown. It can be concluded that the previous model of ANFIS is more efficient and stable in terms of reduced error during training. By comparison epoch number greater than optimization PSO, previously ANFIS can approach the estimation accuracy of ANFIS-PSO.

For future work, similar studies can be done to predict software effort using optimization models based on machine learning algorithms, such as Genetic Algorithms (GA) and Ant Colony Optimization (ACO).

TABLE II Performance Evaluation Result

	Best epoch	Least Error	MMSE	MMRE	PRED(25)
ANFIS Triangular	10	4.94	22.6138566	0.102283	0.9047619
ANFIS-PSO Triangular	107	2.22	9.30948262	0.05998876	0.96825397
ANFIS Gaussian	20	8.15	12.4849286	0.06504668	0.95238095
ANFIS-PSO Gaussian	500	7.04	10.6328251	0.05544703	0.95238095
ANFIS SubClust	2	0.52	9.31011887	0.00651858	1
ANFIS-PSO SubClust	150	0.07	4.09384171	0.0040061	1

References

[1] S. Grimstad and M. Jorgensen, "The Impact of Irrelevant Information on Estimates of Software Development Effort," *Proceedings of the 2007 Australian Software Engineering Conference, ASWEC '07 IEEE Computer Society*, pp. 359-368, 2007.

[2] "http://www.cin.ufpe.br/~gmp/docs/papers/extreme_chaos2001.pdf," 2001. [Online]. Available: http://www.cin.ufpe.br/~gmp/docs/papers/extreme_chaos2001.pdf.

[3] F. Douglas Fisher, Srinivasan and Krishnamoorthy, "Machine learning approaches to estimating software development effort," *IEEE Transactions on Software Engineering vol.21, no.2*, pp. 126-137, 1995.

[4] Prabhakar and Maitreyee Dutta, "Application of machine learning techniques for predicting software effort," *Elixir Comp. Sci. & Engg*, pp. 13677-13682, 2013.

[5] Prabhakar and M. Dutta, "Prediction of Software Effort using Artificial Neural Network and Support Vector Machine," *International Journal of Advanced Research in Computer Science and Software Engineering, vol.3, no. 3*, pp. 40-46, 2013.

[6] C. S. Reddy, P. S. Rao, KVSVN Raju and V. V. Kumari, "A New Approach for Estimating Software Effort Using RBFN Network," *International Journal of Computer Science and Network Security vol.8, no.7*, pp. 237-241, 2008.

[7] P. Reddy, "Software Effort Estimation using Radial Based and Generalized Regression Neural Network," *Journal of Computing, vol.2, no. 5*, pp. 87-92, 2010.

[8] R. Bhatnagar, V. Bhattacharjee and M. K. Ghose, "Software Development Effort Estimation - Neural Network Vs. Regression Modeling Approach," *International Journal of Engineering Science and Technology, vol.2*, pp. 2950-2956, 2010.

[9] Azzeh M., Neagu D. and Cowling P. I., "Fuzzy Grey Relational Analysis for Software Effort Estimation," *Empirical Software Engineering*, pp. 60-90, 2010.

[10] S. Dingra and P. Singh, "An Adaptive Neuro Fuzzy Approach for Software Development Time Estimation," *International Journal of Advanced Research in Computer Science and Software Engineering, vol.3, no. 5*, pp. 586-590, 2013.

[11] Lin J. C., Chang C. T. and Huang S. Y., "Research on Software Effort Evaluation Combined with Genetic Algorithm and Support Vector Regression," *IEEE International Symposium on Computer Science and Society (11)*, pp. 349-352, 2011.

[12] K. Sweta and P. Shashankar, "Comparison and Analysis of Different Software Cost Estimation Method vol.4, no.1," *International Journal of Advanced Computer Science and Applications*, pp. 153-157, 2013.

[13] CH.VM.K. Hari and P. Reddy, "A Fine Parameter Tuning for COCOMO 81 Software Effort Estimation using Particle Swarm Optimization," *Journal of Software Engineering (ISSN: 1819-4311/DOI:*

10.3923/jse.2011), 2011.

[14] Lin J. C. and Tzeng H. Y., "Applying Particle Swarm Optimization to Estimate Software Effort by Multiple Factors Software Project Clustering," *IEEE Computer Symposium (ICS),* pp. 1039-1044, 2010.

[15] X. Huang, D. Ho, J. Ren and L. F. Capretz, "Improving the COCOMO Model using a Neuro-Fuzzy Approach," *Applied Soft Computing, no.7,* pp. 29-40, 2007.

[16] J. S. Pahariya, V. Ravi, M. Carr and M. Vasu, "Computational Intelligence Hybrids Applied to Software Cost Estimation," *International Journal of Computer Information Systems and Industrial Management Applications (IJCISIM) vol.2,* pp. 104-112, 2010.

[17] L. Jin-Cherng Lin, C. Chu-Ting and H. Sheng-Yu, "Research on Software Effort Estimation Combined with Genetic Algorithm and Support Vector Regression," *Computer Science and Society (ISCCS) International Symposium,* pp. 349-352, 2011.

[18] B. Boehm, "Software engineering economics," *IEEE Transactions on Software Engineering,* pp. (10) 4-21, 1984.

[19] B. Boehm, "Safe and Simple Software Cost Analysis," *IEEE Software,* pp. 17(5) 14-17, 2000.

[20] L. H. Putnam, "SLIM: a quantitative tool for software cost and schedule estimation," in *Proceedings of NBS/IEEE/ACM Software Tool Fair,* 1981.

[21] H. A. Rubin, "Macroestimation of software development parameters: The Estimacs System," in *Proceedings of SOFTFAIR Conference on Software Development Tools, Techniques and Alternatives,* Arlington, 1983.

[22] A. J. Albrecht, "Measuring Application Development Productivity," in *Proceedings of IBM Application Development Symp.*, Monterey, California, 1979.

[23] Sheta A.F., Ayesh A. and Rine D., "Evaluating Software Cost Estimation Models using Particle Swarm Optimi-zation and Fuzzy Logic for NASA Projects: a Comparative Study," *International Journal Bio-Inspired Computation, vol.2, no. 6,* pp. 365-373, 2010.

[24] Foss T., Stensrud E., Kitchenham B. and Myrtveit L., "A Simulation Study of the Model Evaluation Criterion MMRE," *IEEE Transactions on Software Engineering vol.29,* pp. 985-995, 2003.

(Check) 26 - Modeling Software Effort Estimation Using Hybrid PSO-ANFIS

ORIGINALITY REPORT

%94	%11	%94	%8
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Suharjito, Saka Nanda, Benfano Soewito. "Modeling software effort estimation using hybrid PSO-ANFIS", 2016 International Seminar on Intelligent Technology and Its Applications (ISITIA), 2016 Publication	%94
2	Submitted to Indian Institute of Technology, Kanpur Student Paper	%1

EXCLUDE QUOTES	ON	EXCLUDE MATCHES	< 5 WORDS
EXCLUDE BIBLIOGRAPHY	ON		